ubiSOAP: A Service Oriented Middleware for Seamless Networking

Mauro Caporuscio, Pierre-Guillaume Raverdy, Hassine Moungla, Valérie Issarny

*ubi*SOAP: A Service Oriented Middleware for Seamless Networking*

Mauro Caporuscio, Pierre-Guillaume Raverdy,
Hassine Moungla, and Valerie Issarny

INRIA Paris-Rocquencourt
Domaine de Voluceau - 78153 Le Chesnay, France
{First.LastName}@inria.fr

Abstract. The computing and networking capacities of today's wireless portable devices allow for pervasive services, which are seamlessly networked. Indeed, wireless handheld devices now embed the necessary resources to act as both service clients and providers. However, the seamless networking of services remains challenged by the inherent mobility and resource constraints of devices, which make services a priori highly volatile. This paper discusses the design, implementation and experimentation of the *ubi*SOAP service-oriented middleware, which leverages wireless networking capacities to effectively enable the seamless networking of services. *ubi*SOAP specifically defines a layered communication middleware that underlies standard SOAP-based middleware, hence supporting legacy services while exploiting nowadays pervasive connectivity.

1 Introduction

With network connectivity being embedded in most computing devices, networking environments are now pervasive. As a result, any networked device may seamlessly consume but also provide software applications over the network. Service-Oriented Computing (SOC) then introduces natural design abstractions to deal with pervasive networking environments [2]. Indeed, networked software applications may conveniently be abstracted as autonomous loosely coupled services, which may be combined to accomplish complex tasks. In addition, the concrete instantiation of SOC paradigms provided by Web Services (WS) technologies by means of Web-based/XML-based open standards (e.g. WSDL, UDDI, HTTP, SOAP) may be exploited for concrete implementation of pervasive services. However, while Web services standards and implementations targeting wide-area domains are effective technologies, supporting Web service access in pervasive networking environments is still challenging. In such kind of networking environments, mobile applications, acting as both service consumers and providers, often run on scarce resource platforms such as personal digital assistants and mobile phones, which have limited CPU power, memory,

* This work is part of the IST PLASTIC project and has been funded by the European Commission, FP6 contract number 026955, http://www.ist-plastic.org/

A. Bouguettaya, I. Krueger, and T. Margaria (Eds.): ICSOC 2008, LNCS 5364, pp. 195–209, 2008.

and battery life. Moreover, these devices are usually interconnected through one or more heterogeneous wireless links, which compared to wired networks are characterized by lower bandwidths, higher error rates, and frequent disconnections. The former issue has led to the introduction of lightweight middleware enabling base WS-oriented communication patterns among wireless portable devices (i.e., SOAP-based messaging and dynamic service discovery) [10,1]. The latter issue has further led to examine alternative SOAP transports [5]. However, a key feature of pervasive networking environments is the diversity of radio links available on portable devices, which may be exploited towards seamless connectivity. Specifically, as nodes get connected via multiple radio links, thorough scheduling and handover across those links allow enhancing overall connectivity and actually making it seamless [23,18,20]. This calls for making services network-agnostic [21], so that the underlying middleware takes care of scheduling exchanged messages over the embedded links in a way that best matches Quality of Service (QoS) requirements [4], and further ensures service continuity through vertical handover [9]. In this setting, a primary requirement for supporting service-oriented middleware is to provide a comprehensive networking abstraction that allows applications to be unaware of the actual underlying networks while still exploiting their diversities in terms of both functional and extra-functional properties.

This paper introduces the *ubi*SOAP communication middleware, which underlies SOAP-based middleware and strives to provide pervasive networking to services. Specifically, *ubi*SOAP defines a two-layer architecture composed of a *multi-radio networking* layer and a *WS-oriented communication* layer, which respectively provide network-agnostic connectivity and SOAP-based unicast and group communication in pervasive networking environments. The design rationale for *ubi*SOAP is further discussed in the next section, while Section 3 details its core constituents. Then, Sections 4 and 5 assess the proposed middleware, discussing respectively *ubi*SOAP usage for implementing an advanced middleware service (i.e., pervasive service discovery), and *ubi*SOAP performance based on experiment. Finally, Section 6 concludes with a summary of our contribution with respect to related work, and our perspectives for future work.

2 Design Rationale

With the drastic evolution of wireless technologies, software services can become truly pervasive, being not solely accessed but also hosted by wirelessly networked portable devices. As a result, legacy applications can become available anytime, anywhere, but also revisited to take full advantage of pervasive networking. Further, new application services may emerge, in particular based on the nomadic feature and ad hoc connectivity of wireless portable devices, as exemplified by emergency rescue scenarios, where mobile portable hosts serve sensing the environment and coordinating rescue actions. Still, enabling pervasive service provisioning on mobile hosts requires special care, as resources

are far more constrained than resource-rich Internet servers originally targeted by service oriented computing and its Web Service instantiation. Further, the mobility of wireless hosts requires special attention. Indeed, early solutions introduced towards nomadic computing targeted service hosts with which connectivity can eventually be restored, while this cannot be assumed in general when services are hosted by mobile devices that connect in an ad hoc way [24]. Overcoming resource constraints of wireless devices in the support of Web services has led to the introduction of custom SOAP engines, among which the open source iCSOAP engine from INRIA, which was developed as part of the WSAMI middleware[1] featuring Web services for ambient intelligence [10].

Regarding mobility issues, portable devices now embed multiple radio interfaces, which may be combined to bring seamless networking to mobile applications [21]. Specifically, embedded networking technologies differ from several respects, among which range, latency, bandwidth, energy consumption, financial cost, availability and so on. Therefore, the scheduling of communications over embedded interfaces according to application requirements can significantly increase the overall QoS. The collection of wireless technologies may in particular be considered as hierarchical *wireless overlay networks*, which are structured according to respective coverage [12]. Then, the interface used for communication may simply depend on the location of the target host. However, different network parameters must be taken into account in the scheduling of communications, so as to meet applications' QoS requirements, while optimizing the overall resource usage [18]. In particular, saving energy is critical for enhanced autonomy of hosts and thus requires selecting as far as possible the network interface that consumes the least energy among those eligible [20,4]. Further, vertical handover across heterogeneous wireless networks must be supported so as to maintain connectivity with nodes despite their mobility across networks [23], and thus bring seamless connectivity to/from mobile nodes.

Among one of its major goals, the *ubi*SOAP communication middleware aims at effectively using the diverse networking technologies in the handling of service-oriented communication, hence offering network-agnostic connectivity to/from nodes. As discussed above, this requires addressing a number of critical issues such as *network availability*, *user and application QoS requirements* and *vertical handover*. The latter issue is particularly important with respect to the *service continuity* requirement. In fact, when switching from a given network to one of a different type, the device is required to change its status according to the new environment it is entering. Indeed, changing the device's status affects also the status of all the devices that are currently interacting with it. Specifically, in an all-IP networking environment, the IP address meaning is twofold: end point identification (i.e., an IP address uniquely identifies a host in a given network) and location identification (i.e., the network in which the host is located). Hence, when a host changes its point of attachment (vertical handover between two networks), the IP address must be modified (i.e., the internal status) accordingly in order to route packets to the new network. Then, since the IP address is

[1] `http://www-rocq.inria.fr/arles/download/ozone/`

the base of any application-layer connection, all the ongoing connections break (i.e., the handover affects the status of the interacting parties). Furthermore, as devices can bind various networks at the same time, two interacting parties might communicate through multiple paths. Hence, choosing the best connection path to serve a given interaction is a key issue to deal with in pervasive networks, as this significantly affects the QoS at large (e.g., availability, performance with respect to both resource consumption and response time, security) [3].

Multi-radio connectivity further allows deploying bridges in the network, effectively realizing a multi-network service overlay [6]. Specifically, resource-rich nodes (e.g., laptops or even reachable stationary nodes) that embed multiple radio interfaces may act as bridges that route messages across heterogeneous networks. Such a feature is beneficial for pervasive services, enabling to overcome the resource limitation and mobility of nodes and contributing to achieve seamless service connectivity. Indeed, this allows energy-limited devices to use the least consuming interface while being able to reach all the devices of the overlay. Further, a networked service that changes physical network following host mobility may still be reachable in the overlay.

Another of our goals for *ubi*SOAP is to support legacy Web services and thus transparently bring the added value of today's pervasive networking environments to existing services. This has in particular led us to layer *ubi*SOAP as a specific transport for SOAP engines (e.g., Axis2, iCSOAP) and to leverage WS-addressing to integrate multi-radio, multi-network connectivity in SOAP headers. In this context, it is crucial to examine carefully the performance of SOAP transports. In particular, it has been shown that the performance of default SOAP over HTTP is poor in wireless environments, further leading to study alternative transports such as TCP and UDP [14,5]. While SOAP over UDP clearly offers the best response time, SOAP over TCP has the advantage of built-in reliability and is further suitable for applications with short requests. *ubi*SOAP thus realizes SOAP-over-TCP unicast messaging as a tradeoffs solution, while integrating SOAP over UDP is an area for future work. Still, another SOAP transport that is of much interest for pervasive networking environments is multicast group communication. Indeed, group-based interactions are central in a number of pervasive computing scenarios, due to the user-centric nature of pervasive computing and the innate group interaction skills of people [7,22]. *ubi*SOAP thus features a base SOAP transport for group communication.

3 *ubi*SOAP Middleware for Pervasive Services

Following the above discussion, the architecture of *ubi*SOAP layers the following constituents below SOAP-based middleware functionalities (see Fig. 1): (*i*) multi-radio networking provides network-agnostic connectivity (see § 3.1), (*ii*) multi-network routing implements a multi-network overlay (see § 3.2), and (*iii*) point-to-point and group transports leverage multi-radio, multi-network message routing, and further introduce communication primitives targeted at pervasive computing systems (see § 3.3).

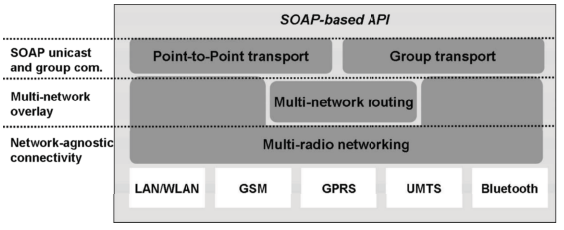

Fig. 1. *ubi*SOAP software architecture

3.1 Network-Agnostic Service Connectivity

The *multi-radio networking layer* of *ubi*SOAP provides core functionalities to effectively manage multi-radio connectivity by providing: (*i*) a network-agnostic addressing scheme together with (*ii*) QoS-aware network link selection.

Network-agnostic addressing. Devices embedding multiple network interfaces may have multiple IP addresses, at least one for each active interface. Thus, in order to identify uniquely a given application in the network we associate to it a *Multi-Radio Network Address* (MRN@). The MRN@ of an application instance is specifically the application's Unique ID, which resolves into the actual set of IP addresses (precisely, $network\_ID \oplus IP$ addresses) bound to the device (at a given time) that runs the given instance (see Fig. 2). Then, upper layers shall use MRN@ as part of their addressing scheme (e.g., through WS-addressing in the case of Web services), which replaces the traditional IP-based addressing scheme. MRN@s are automatically generated and managed by the multi-radio networking layer. Furthermore, the multi-radio networking layer allows for performing a lookup operation that, starting from an MRN@, returns the set of IP addresses actually bound to it. The basic operations provided by the multi-radio networking layer are as follows. First, *Registration* allows the application to register within the multi-radio networking layer and generates the MRN@ that uniquely identifies it. In particular, the user application provides as input an identifier (locally unique), which is used to generate the MRN@ to be returned. Then, *Lookup* allows user applications to retrieve the actual set of IP addresses related to a given MRN@. If the resolution of MRN@ is not cached or needs to be updated, a request is multicasted to all the networks currently accessible and, if the device related to such MRN@ is reached, it will directly reply to the requester by supplying the actual set of IP addresses. Base unicast and multicast communication schemes are provided on top of MRN@ network-agnostic addressing: (*i*) *Synchronous unicast* is provided by means of a packet input/output stream that is used to read/write packets to be exchanged during the interaction between client and server applications; and (*ii*) *Asynchronous multicast* allows the user application to send multicast packets to all members of a given group.

QoS-aware interface activation and network selection. Next to MRN@ addressing, it is crucial to activate and select the best possible networks (among

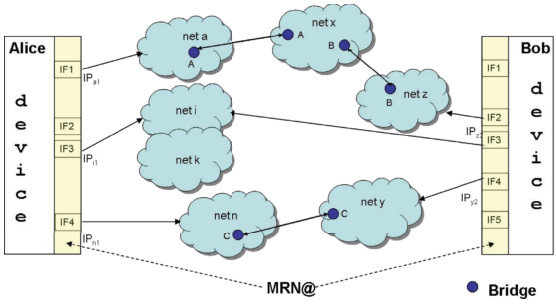

Fig. 2. *ubi*SOAP multi-radio multi-network connectivity using MRN@ and bridging

those available) with respect to required QoS. *Interface activation* allows the user application to activate the best possible interfaces (among those available) with respect to the required QoS. In particular, the application submits its QoS requirement (specified as set of pairs $< QoS_{attribute}, QoS_{value} >$) to the multi-radio networking layer, which in turn compares it with the QoS of each available interface. In this case, since the interface is switched off, QoS refers to the theoretic values of a network interface declared by the manufacturer (e.g., GPRS maximum bitrate = 171.2Kb/s). If the interface satisfies the requirement posed by the application, within a given approximation expressed in percentage, it is activated. It is also possible to define priorities upon the various quantitative parameters, in order to specify if a given parameter is more important than the others. *Network selection* is performed during the establishment of the communication and takes into account the QoS attributes required by the client application that is initiating the connection, as well as the networks active on the server listening for incoming connections, as given by the server's MRN@. If the client and the server share only one network that satisfies the requirements, it is used to carry on the interaction. On the other hand, when the two parties share more than one network, the selection algorithm selects the one that best meets the required QoS.

3.2 Multi-network Service Overlay

Thanks to the *ubi*SOAP multi-radio networking layer, communication among nodes exploits the various network links that the nodes have in common, further selecting the link that provides the required QoS. However, in some cases, it might also be desirable for nodes to be able to access services that are hosted in networks that the requesting node is not directly connected to (e.g., to provide continuity of service despite node mobility). For this purpose, *ubi*SOAP introduces an overlay network that bridges heterogeneous networks, thus enhancing overall service connectivity. Specifically, nodes that are connected to two (or more) different networks through their network interfaces can assume the role of bridge nodes. Bridge nodes quite literally "bridge" between two separate networks, relaying point-to-point and multicast *ubi*SOAP messages across those

networks. Still, we assume that nodes will not request services that would require the consecutive traversal of more than five wireless networks (see [8,15] for a detailed analysis on wireless communication) in order to access them.

Multi-network point-to-point routing. In *ubi*SOAP multi-network, multi-radio environments, the $network\_ID \oplus IP$ address embedded in the MRN@ of a host contains, along with the network address of the service host, the network ID that uniquely identifies the network (e.g., a BSSID, MAC address of the Bluetooth master, etc.) that the host resides in under its given address. For instance, in Fig. 2, the of device Alice is connected to networks `a`, `i`, and `n`, through its various network interfaces. Clearly, the device can trivially access services hosted in these networks. However, in order to access services hosted in the distant networks `x`, `y`, and `z`, the device has to route its request through an appropriate bridge node (i.e., Bridges `A`, `B` and `C`, noting that each bridge node is displayed in each network it is part of). To achieve effective routing across bridges, we propose a straightforward approach based on the principle of Mobile Ad hoc NETwork (MANET) routing. In this approach, bridge nodes advertise their presence to the nodes in their corresponding networks and exchange routing information. For this purpose, bridge nodes run an instance of OLSR [11] among each other. Instead of concrete node addresses, however, bridges store as destinations the identifiers of the various present networks (i.e., $network\_ID$) and as next hop the bridge that needs to be contacted next to eventually reach the target network. Being a proactive routing protocol, this inter-bridge OLSR instance gives each bridge the required routing information to reach all connected networks. Whenever a non-bridge node wants to access a service outside one of the networks it is itself connected to, it may simply route the request to any bridge of choice that will then forward the request accordingly. As mentioned above, bridge nodes periodically advertise their presence to the nodes in their respective networks. As a further optimization, bridge nodes may include in their advertisements their OLSR routing tables so that non-bridge nodes may choose bridge nodes according to metrics such as network hops, etc.

Multi-network multicast routing. It is crucial to support both point-to-point and group interactions in the multi-radio, multi-network environment. In particular, multicasting is central to advanced middleware services like dynamic discovery [25]. We thus introduce multi-network multicast routing, building upon multicast facilities of the composed networks. The base principles of the *ubi*SOAP multi-network multicast routing are depicted in Fig. 3; multicast routing is such that within an IP network, the network's multicast facility (i.e., IP multicast or higher level group communication like Java Groups) is used for communication among group members. Then, multicast messages are forwarded by *ubi*SOAP bridges up to a fixed number of hops (i.e., 5 as discussed previously), while avoiding cycles and duplication. The *ubi*SOAP multi-network routing facility enables the definition of application-level groups. Specifically, application-level groups are individually managed at the *ubi*SOAP middleware layer while a single *ubi*SOAP multi-network group is managed in the network layer. The latter

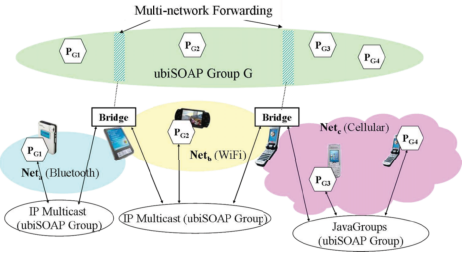

Fig. 3. *ubi*SOAP multi-network multicast routing

is actually a composition of *ubi*SOAP multicast groups, one for each of the composed networks.

3.3 Custom SOAP Transports for Pervasive Services

In order to leverage the provided multi-radio, multi-network service connectivity, *ubi*SOAP introduces custom SOAP transports. Specifically: (*i*) the provided SOAP transport for point-to-point communication brings multi-radio multi-network routing to legacy SOAP messaging, while (*ii*) the SOAP transport for group communication enhances the SOAP API to meet the corresponding base requirement of pervasive networking environments [13,7,5,22,25].

Point-to-point communication. The *ubi*SOAP point-to-point transport is a connection-oriented transport for supporting communication between a client and a service. This transport interacts with the multi-radio networking layer to send and receive messages over the network based on the MRN@ that identifies the remote party. It also interacts with the SOAP engine or the client SOAP library to receive or dispatch SOAP messages locally. When sending a message, the *ubi*SOAP point-to-point transport must first evaluate if the destination is directly reachable (i.e., the MRN@ of the sender and of the destination share a common network). If true, the message is then sent directly to the destination. If not, the transport retrieves the MRN@ of a *ubi*SOAP bridge directly reachable, encapsulates as plain data the application's SOAP message into a specific forwarding message, and sends this forwarding message to the bridge. This message is forwarded between bridges until it reaches the destination where the application's SOAP message is extracted and dispatched. While the client blocks until the response is received, the forwarding message is routed between *ubi*SOAP bridges using connectionless communication. The response message may thus follow a different route. The first bridge returns the response to the client, or terminates the connection after a given timeout.

On the service side, the *ubi*SOAP point-to-point transport interacts with the SOAP engine to deliver the message to the appropriate service. For messages that have been routed by bridges, the destination must extract the actual SOAP message, and also set up properly the SOAP message parameters (i.e., URL

of the service, action to perform). This is achieved by storing (on the source side) and retrieving (on the destination side) the relevant information in the *ubi*SOAP header of the SOAP message. In particular, the set of IP addresses associated to an MRN@ is embedded in the header of request (for the client) and response (for the service) messages. This enables communicating devices in different networks to keep track of mobile nodes and maintain sessions (as long as a communication path exists). In some cases however (i.e., when both the client and the service simultaneously change the complete set of IP addresses associated to their MRN@), and no direct link exists), the session will close and the client will need to perform a service discovery (See Section X) to find the same service again and restart the communication.

Group communication. The *ubi*SOAP group transport is a connectionless transport for one-way communication between multiple peers in multi-network configurations. The *ubi*SOAP group transport component interacts with the multi-radio networking layer to send group messages based on an MRN@ identifying the group, and with the SOAP engine to deliver the group's messages to the registered services. As noted above, groups are identified with an MRN@. Multicast-based applications usually assume that all group members agree beforehand on a specific IP address for the group. We therefore also assume that all group members use the same MRN@ for the group. While services are not able to directly return a result to a client (one-way multicast), a service may send a message (one-way unicast) on the group directed at a specific peer (i.e., similar to the *sendTo* socket call). As group communication in the underlying multi-radio networking layer is multicast-based, it does not guarantee the ordering or the delivery of messages. While ordering may be easily achieved on the receiving side, the overhead to provide group reliability is deemed too costly due to the dynamics of pervasive networks. Also, while many mobile devices may run the same collaborative application, a user may only be interested in interacting with the ones at its location. Such scoping may be achieved by limiting the forwarding of group's messages or by adding forwarding constraints.

4 *ubi*SOAP in Action: Pervasive Service Discovery

*ubi*SOAP is being developed as part of a larger initiative on assisting the development of dependable services for pervasive networking environments, which is undertaken by the European IST PLASTIC project[2]. The PLASTIC project specifically investigates the development of the PLASTIC platform, decomposing into a development environment, service-oriented middleware and validation framework for the target pervasive services. *ubi*SOAP then defines the PLASTIC core middleware while advanced middleware services are being developed on top of it to address the requirements of pervasive networking (i.e., dynamic service discovery and composition, security and context management). Further, the PLASTIC platform is being assessed against actual case studies in the area

[2] `http://www.ist-plastic.org/`

of eHealth, eLearning, eBusiness and eVoting. This in particular allows us to extensively experiment with the *ubi*SOAP middleware. In this section, we focus on one use of *ubi*SOAP that is realizing dynamic discovery of pervasive services, which benefits from all the advanced features of *ubi*SOAP, i.e., group communication, and multi-radio, multi-network message routing.

Service discovery is an essential function of SOC as it enables the runtime association to the networked services. Three basic roles are identified for service discovery: (*i*) Service provider is the role assumed by a software entity offering a networked service; (*ii*) Service requester is the role of an entity seeking to consume a specific service; (*iii*) Service repository is the role of an entity maintaining information on available services and a way to access them. A service description formalism or language to describe the functional and non-functional properties (such as QoS, security or transactional aspects of networked services) complemented with a service discovery protocol enables service providers, requesters and repositories to interact with each other. Many academic and industry supported SDPs have already been proposed and leading SDPs in dynamic environments use a pull-based approach (SLP, WS-Discovery, Jini, SSDP), often supporting both the centralized and distributed modes of interaction: clients send requests to service providers (distributed pull-based mode) or to a third-party repository (centralized pull-based mode) in order to get a list of services compatible with the request attributes.

Building on the tremendous number of proposed service discovery protocols and accounting for the specifics of pervasive computing [25], we introduce a Pervasive Service Discovery (PSD) Service that provides dynamic, interoperable, context-aware service discovery. PSD is mainly a reengineering of the open source MUSDAC multi-protocol service discovery platform[3] [19] on top of *ubi*SOAP in order to support service discovery in multi-radio, multi-network environments. PSD uses a hierarchical approach for service discovery in multi-network environments (see Fig. 4). Indeed, a (logically) centralized repository (PSD-S) coordinates service discovery within an independent network, while PSD-Ss in different networks communicate together in a fully distributed way to disseminate service information. While in MUSDAC service discovery and access were tightly integrated, PSD-Ss are only concerned with service discovery, and rely on *ubi*SOAP group communication to disseminate service information across networks. Changes in the multi-network topology (e.g., broken propagation paths) are then taken care of transparently. PSD-Ss (see Fig. 4) provide an explicit API supported by the *PSD plugin* that enables clients (resp. providers) in a network to discover (resp. advertise) a service in the multi-network environment. It further enables clients and providers to benefit from advanced discovery features (e.g., context-awareness) by directly issuing requests or advertisements in the PSDL format. Specific *legacy SDP plugins* register with the active SDPs in the network, and translate requests and advertisements in legacy formats to PSDL (e.g., SLP and UPnP in Fig. 4), which are stored in the *PSD repository*. The *matching engine* then combines various matching algorithms [17] to support

[3] `http://www-rocq.inria.fr/arles/download/ubisec/`

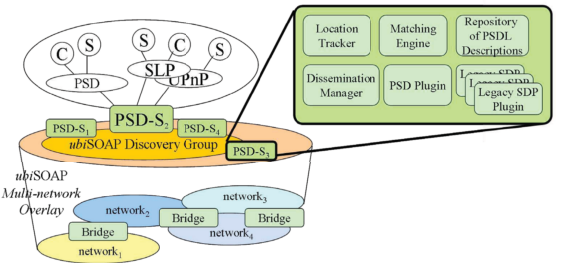

Fig. 4. Pervasive Service Discovery

the various elements of the service description (for both requests and advertisements), and thus provides comprehensive interoperability between SDPs. Finally, the *dissemination manager* controls the dissemination of local requests and the compilation of the results returned by distant PSD-Ss, while the *location tracker* collaborates with lower-level services in the *ubi*SOAP middleware to maintain the physical address of mobile services discovered in the environment.

As described above, the PSD repository stores PSDL service descriptions that are either generated by legacy SDP plugins (e.g., UPnP2PSD plugin) or directly registered by service providers using the PSD plugin. We use a hierarchical service description format that actually combines a number of distinct documents specifying different facets of the service. The PSDL description acts primarily as a top-level container for additional files describing facets of the service. For example, a WSDL document may be used to describe the service interface while non-functional properties can be described using existing QoS and context models. The *ubi*SOAP grounding of host, that identifies the networks and IP address at which the service's host is network-reachable (i.e., MRN@ and mapping) is described in such separate document thus facilitating dynamic updates.

5 Experimentation

We have implemented a prototype of the *ubi*SOAP middleware using Java for both desktop (J2SE) and mobile (J2ME CDC) environments. As mentioned previously, the *ubi*SOAP prototype is being extensively experimented with as part of the PLASTIC project, and has been released under open source license[4]. To assess the efficiency of the *ubi*SOAP middleware, we evaluate the processing time to call a simple Echo Web service under various network configurations. We evaluate both the time required to call the Web service the first time, which includes the dynamic service creation/instantiation, and the time required to call the Web service after it is deployed. Tests are performed on a Windows XP PC with a 2.6GHz processor and 1 GB of memory for the desktop platform, and on

[4] *ubi*SOAP is composed of the Multi-radio Networking and B3GSOAP packages available at `http://www.ist-plastic.org/`

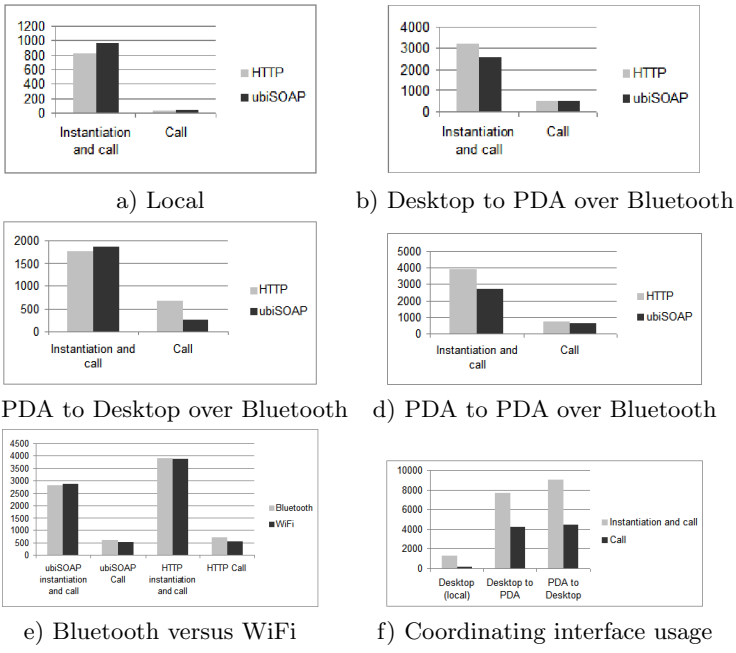

Fig. 5. *ubi*SOAP performance

a HP iPaq hw6910 (Intel PXA 270 at 416 MHz) and a HP iPaq 110 (PXA310 at 624 MHz) for the mobile platforms. We further use IBMs J9 JVM (J2ME CDC 1.1), and the open source iCSOAP lightweight SOAP engine[5]. Results presented are the average of 5 runs with 100 call each.

Figures 5.a) to d) assess the performance in ms, of *ubi*SOAP unicast transport versus HTTP in the following configurations: a) both client and service provider running on the desktop platform, b) client running on desktop and service provider on PDA, c) client running on PDA and service provider on desktop, and d) client and server running each on a PDA. Provided results subdivide into the response time of the initial call including the service instantiation (i.e., *Instantiation and call*), and the average response time of subsequent calls (i.e., *Call*). Configuration a) shows that the time taken for service instantiation far exceeds the one of calls. The response time of *ubi*SOAP is further slightly higher than the one of HTTP, which is due to the management of MRN@ whose processing gets noticeable due to the fact that the communication is local. Configuration b) then demonstrates that *ubi*SOAP outperforms HTTP by about 25% at instantiation time. This is explained by the fact that the processing overhead (header and session management) is much lighter in *ubi*SOAP, and the difference in processing is emphasized by the limited capacities of the devices (slow task switching, memory access, ...). As shown by Configuration c), in the case where the service provider runs on the desktop, HTTP performance

[5] Available at `http://www-rocq.inria.fr/arles/download/ozone/`

for instantiation is slightly better, because the difference in the management of session is negligible on the desktop, while *ubi*SOAP on the desktop adds the cost of MRN@ resolution through multicast. On the other hand, the *ubi*SOAP performance for calls is better, again due to lightweight session management on the client side and the fact that the MRN@ needs only to be resolved at the first call. Finally, Configuration d) aggregates the above results b) and c), showing the enhanced performance of *ubi*SOAP over HTTP when both client and service provider run on a PDA.

Figure 5.e) complements the above measures by showing the performance of wireless communications between PDAs using both WiFi and Bluetooth. The higher bandwidth of WiFi makes it obviously better positioned for handling communication compared to Bluetooth, even for small size messages. However, this takes into account performance only, while other criteria are of relevance like power consumption [4], as addressed by the QoS-aware network selection realized by the *ubi*SOAP multi-radio networking layer. Although the theoretical bandwidth for Bluetooth and WiFi is significantly different, the actual bandwidth available to applications depends on the hardware, drivers, and, in our case, the Java JVM ability to cope with the load. As demonstrated in the experiments, the actual bandwidth for Java applications on PDA is almost identical for Bluetooth and WiFi.

In general, the scheduling of communication over network interfaces shall account for the QoS requirements of networked applications. Such requirement and specifically conflicts between different applications should be taken into account when activating/desactivating network interfaces or performing handover. We thus have implemented a version of *ubi*SOAP, which deals with coordinated usage of the network interfaces, so that the actual network used for communication is selected according to the applications that are currently run. Specifically, a daemon process is introduced, which coordinates the usage of the network interfaces. However, such a solution suffers from the resource availability of PDAs that is still currently limited, and accommodates poorly the concurrent execution of applications. Indeed, as shown by results of Figure 5.f) that provides the response time of *ubi*SOAP with the daemon running, while the overhead on the desktop is reasonable, it increases dramatically when performed on the PDA due to the constant process switching.

6 Conclusion

Service-oriented computing appears as a promising paradigm for pervasive computing systems that shall seamlessly integrate the functionalities offered by networked resources, both mobile and stationary, both resource-rich and resource-constrained. In particular, the loose coupling of services makes the paradigm much appropriate for wireless, mobile environments that are highly dynamic. However, enabling service-oriented computing in pervasive networking environments raises key challenges among which overcoming resource constraints and volatility of wireless, mobile devices. This has in particular led to introduce

lightweight service-oriented middleware [10,2,1,16]. However, to the best of our knowledge, none of the existing solutions comprehensively integrate the full capacity of today's pervasive networking environments, which allow wireless devices to interact via multiple network paths. Such a feature actually enables seamless networking and in particular overcoming the nodes' mobility through vertical handover across networks. This further allows for tuning network usage according to application requirements, and thus enhancing overall QoS.

Exploiting multi-radio connectivity has led to the definition of various algorithms for optimizing the scheduling of communications over multiple radio interfaces, e.g., [12,23,18,20,4]. Building on this effort, this paper has introduced *ubi*SOAP, which implements SOAP transports that leverage multi-radio, multi-network connectivity and may be coupled with lightweight engines like iCSOAP [10]. As a result *ubi*SOAP enables the seamless networking of Web services that may be deployed on various devices, including mobile devices like today's smart phones embedding multiple radio interfaces. *ubi*SOAP is now being extensively experimented as part of the PLASTIC European project, being the basis for the development of various pervasive services, ranging from middleware-layer pervasive service discovery to application-layer services in the area of eBusiness, eHealth, eLearning and eVoting. Experiment further shows that the performance of *ubi*SOAP are in general better than default SOAP-over-HTTP transport, thanks to lightweight session management.

Our current work relates to the aforementioned experimentation of *ubi*SOAP for thorough assessment prior to its release under open source license. Our future work relates to further evolution of *ubi*SOAP to meet the numerous requirements of pervasive computing. Part of this effort lies in introducing additional SOAP transports, like an UDP-based one, which significantly improves response time, while affecting reliability [14]. Obviously, *ubi*SOAP needs to be complemented with a number of middleware services to deal with QoS; such an issue is addressed as part of the PLASTIC project where middleware solutions for security and context awareness are being investigated. We are also examining the coupling with semantic-based solutions to enable more precise, yet more flexible, descriptions of pervasive services, as introduced in, e.g., [16].

References

1. Aijaz, F., Hameed, B., Walke, B.: Towards peer-to-peer long lived mobile Web services. In: Proc. of IIT (2007)
2. Bellur, U., Narendra, N.C.: Towards service orientation in pervasive computing systems. In: Proc. of ITCC (2005)
3. Caporuscio, M., Charlet, D., Issarny, V., Navarra, A.: Energetic performance of service-oriented multi-radio networks: issues and perspectives. In: Proc. of WOSP (2007)
4. Charlet, D., Issarny, V., Chibout, R.: Energy-efficient middleware-layer multi-radio networking: An assessment in the area of service discovery. Comput. Netw. 52(1) (2008)
5. Gehlen, G., Aijaz, F., Walke, B.: Mobile Web service communication over UDP. In: Proc. of VTC (2006)

6. Grace, P., Coulson, G., Blair, G.S., Porter, B.: Addressing network heterogeneity in pervasive application environments. In: Proc. of InterSense (2006)
7. Grudin, J.: Group dynamics and ubiquitous computing. Com. ACM 45(12) (2002)
8. Gupta, P., Kumar, P.: The capacity of wireless networks. IEEE Transactions on Information Theory 46(2) (2000)
9. Huang, H., Cai, J.: Improving TCP performance during soft vertical handoff. In: Proc. of AINA (2005)
10. Issarny, V., Sacchetti, D., Tartanoglu, F., Sailhan, F., Chibout, R., Levy, N., Talamona, A.: Developing ambient intelligence systems: A solution based on Web services. Journal of Automated Software Engineering 12(1) (2005)
11. Jacquet, P., Mühlethaler, P., Clausen, T., Laouiti, A., Qayyum, A., Viennot, L.: Optimized link state routing protocol for ad hoc networks. In: Proc. of INMIC (2001)
12. Katz, R.H., Brewer, E.A.: The case for wireless overlay networks. In: Mobile Computing. Kluwer Academic Publishers, Dordrecht (1996)
13. Kindberg, T., Fox, A.: System software for ubiquitous computing. IEEE Pervasive Computing Magazine 1(1) (2002)
14. Lai, K.Y., Phan, T.K.A., Tari, Z.: Efficient SOAP binding for mobile web services. In: Proc. of LCN (2005)
15. Li, J., Blake, C., Couto, D.S.J.D., Lee, H.I., Morris, R.: Capacity of ad hoc wireless networks. In: Proc. of MobiCom (2001)
16. Mokhtar, S.B., Preuveneers, D., Georgantas, N., Issarny, V., Berbers, Y.: Easy: Efficient semantic service discovery in pervasive computing environments with QoS and context support. J. Syst. Softw. 81(5) (2008)
17. Mokhtar, S.B., Raverdy, P.-G., Urbieta, A., Cardoso, R.S.: Interoperable semantic & syntactic service matching for ambient computing environments. In: Proc. of AdhocAmC (2008)
18. Qureshi, A., Guttag, J.: Horde: separating network striping policy from mechanism. In: Proc. of MobiSys (2005)
19. Raverdy, P.-G., Riva, O., de La Chapelle, A., Chibout, R., Issarny, V.: Efficient context-aware service discovery in multi-protocol pervasive environments. In: Proc. of MDM (2006)
20. Sorber, J., Banerjee, N., Corner, M.D., Rollins, S.: Turducken: hierarchical power management for mobile devices. In: Proc. of MobiSys (2005)
21. Su, J., Scott, J., Hui, P., Crowcroft, J., de Lara, E., Diot, C., Goel, A., Lim, M., Upton, E.: Haggle: Seamless networking for mobile applications. In: Krumm, J., Abowd, G.D., Seneviratne, A., Strang, T. (eds.) UbiComp 2007. LNCS, vol. 4717, pp. 391–408. Springer, Heidelberg (2007)
22. Wang, B., Bodily, J., Gupta, S.K.S.: Supporting persistent social groups in ubiquitous computing environments using context-aware ephemeral group service. In: Proc. of PerCom (2004)
23. Wang, H.J., Katz, R.H., Giese, J.: Policy-enabled handoffs across heterogeneous wireless networks. In: Proc. of WMCSA (1999)
24. Zarras, A., Fredj, M., Georgantas, N., Issarny, V.: Engineering reconfigurable distributed software systems: Issues arising for pervasive computing. In: Butler, M., Jones, C.B., Romanovsky, A., Troubitsyna, E. (eds.) Rigorous Development of Complex Fault-Tolerant Systems. LNCS, vol. 4157, pp. 364–386. Springer, Heidelberg (2006)
25. Zhu, F., Mutka, M.W., Ni, L.M.: Service discovery in pervasive computing environments. IEEE Pervasive Computing 4(4) (2005)